VDC Planner: Dynamic Migration-Aware Virtual Data Center Embedding for Cloud

Mohamed Faten Zhani*, Qi Zhang*, Gwendal Simon† and Raouf Boutaba*

*University of Waterloo, Canada
†Telecom Bretagne, Institut Mines Telecom, France

IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)
Ghent, Belgium
27-31 May 2013
Outline

• Introduction
• VDC Planner architecture
• Problem formulation
• Proposed algorithms
• Experiments
• Conclusions
• Future work
Introduction

- Currently, cloud providers provide only computing resources but do not provide any guaranteed network resources.
- Goal: Providing both guaranteed computing and network resources.
  - Virtual Data Centers (VDCs): virtual machines, routers, switches, and links.
Objectives

- Map VDCs onto physical infrastructure (Computing + networking resources)
- Maximize acceptance ratio/revenue
- Minimize energy costs
- Minimize the scheduling delay
- Achieve all of the above objectives dynamically over-time

Our solution: VDC Planner

- A migration-aware virtual data center embedding framework
- VDC embedding, VDC scaling
- Dynamic VDC consolidation.
Possible scenarios

- VDC 1: Initial Embedding
- VDC 2: Scaling up
- VDC 3: Scaling down

Dynamic VDC Consolidation
VDC planner Architecture

Figure 1: VDC Planner Architecture
Problem formulation

- **Objective function**

\[
\min \sum_{\bar{n} \in \bar{N}} y_{n}\bar{p}_{n} + \sum_{i \in I} \sum_{n \in N_i} \sum_{\bar{n} \in \bar{N}} \gamma_{n} x_{n\bar{n}}^i g_{n\bar{n}}^i
\]

- **Operational costs**
- **embedding cost**

- \(y \downarrow n\) a Boolean that indicates that \(n\) is active
- \(x \downarrow nn \uparrow i\) a Boolean that indicates that \(n\) is embedded in \(n\)

- **The embedding cost**

\[
g_{n\bar{n}}^i = \begin{cases} 
mig(n, \bar{m}, \bar{n}) & \text{if } \bar{n} \neq \bar{m} \\
0 & \text{if } \bar{n} = \bar{m} \\
0 & \text{if } n \text{ is currently not embedded} \end{cases}
\]

- **Placement constraint**

\[
x_{n\bar{n}}^i \leq \tilde{x}_{n\bar{n}}^i \quad \forall i \in I, n \in n, \bar{n} \in \bar{N}
\]

\[
\bar{x}_{nn}^i = \begin{cases} 
1 & \text{if node } n \text{ of VDC } i \text{ can be embedded in } \bar{n} \\
0 & \text{otherwise} \end{cases}
\]
**Migration-Aware VDC Embedding Heuristic**

- Sort the VMs by their size
  
  \[
  \text{size}_n^i = \sum_{r \in R} w^r c_n^{ir}
  \]

- Compute the embedding cost (for each VM and physical node)
  
  \[
  \text{cost}_n^i(n, \bar{n}) = \gamma_n (\text{mig}(n, \bar{m}, \bar{n}) + \text{MigOther}(n, \bar{n})) + \sum_{n' \in N^i : (n', n) \in L^i} d(n', \bar{n}) \cdot b(n', n)
  \]  
  \[\text{Equation (19)}\]

- Embed the VM in the physical machine with the minimal embedding cost
Dynamic VDC Consolidation Algorithm

• Sort the physical nodes in increasing order of their utilizations

\[ U_n = \sum_{r \in R} \sum_{i \in I} \sum_{n \in N} \frac{w^r c^i r}{c^r_n}, \]

• Migrate the VMs hosted in low-utilization machines (using Algorithm 1)

• If all VMs are successfully migrated, the machine is turned off.
Experiments

- Physical data center:

  - 4 core switches
  - 4 aggregation switches
  - 4 top-of-rack switches
  - 400 physical machines (8 Cores, 8GB, 100 GB disk).

The VL2 Topology
(Greenberg et al., 2009)
Experiments

• VDC requests:
  o Number of VMs/VDC: [1-20]
  o VM requirements:
    • 1 – 4 cores
    • 1 – 2GB of RAM
    • 1 – 10GB of disk space
  o Virtual link capacity: [1-10 Mbps]
  o Arrival: Poisson distribution
    • 0.01 request/second during night time
    • 0.02 request/second during day time
  o VDC lifetime: exponential distribution (~3 hours)
  o Maximum waiting time: 1 hour
Experiments

• Comparison metrics:
  o Gain in acceptance Ratio
  \[ A_{m/n} = 100 \times \frac{A_m}{A_n} - 100 \]
  o Gain in revenue
  \[ G_{m/n} = 100 \times \frac{R_m}{R_n} - 100 \]
  o Gain in number of active machines
  \[ M_{m/n} = 100 \times \frac{M_m}{M_n} - 100 \]
  o Request scheduling delay
Migration-aware Embedding vs. Baseline

(a) Instantaneous income gains

(b) Gain in acceptance ratio

(c) Gain in terms of queuing delay
Migration-Aware embedding + Consolidation

(a) Migration-aware algorithm (Revenue gain up to 17%)

(b) Migration-aware embedding + consolidation (Revenue gain up to 15%)
Conclusions

• The migration-aware embedding can lead to a gain in terms of revenue and acceptance ratio that can reach up to 17%

• Combined with consolidation, VDC planner uses up to 14% less machines than the Baseline.

• Reduce the scheduling delay by up to 25%.
Future work

• Conduct experiments with real traces/real testbed.

• Combine the Migration-Aware embedding with a capacity provisioning technique
  o The provisioning technique provides the optimal number of machines to be turned on.
  o The migration-Aware embedding will maximize the utilization and the revenue
Thank you
Related Work

• SecondNet [8] is a data center network virtualization architecture
  o a greedy heuristic for VDC embedding problem

• Oktopus [1] proposed two abstractions (virtual cluster and virtual oversubscribed cluster)
  o A greedy heuristic for VDC embedding in tree-like topologies

• SecondNet and Oktopus do not consider migration
Migration-Aware VDC Embedding Heuristic

**Algorithm 1** Algorithm for embedding VDC request $i$

1. Sort $\bar{N}$ based on their states (active or inactive)
2. $S \leftarrow N^i$
3. repeat
4. Let $C \subseteq S$ be the nodes that are connected to already embedded nodes
5. if $C = \emptyset$ then
6. Sort $S$ according $size^i_n$ defined by equation (18).
7. $n^* \leftarrow$ first node in $S$
8. else
9. Sort $C$ according $size^i_n$ defined by equation (18).
10. $n^* \leftarrow$ first node in $C$
11. end if
12. for $\bar{n} \in \bar{N}$ in sorted order do
13. Compute embedding cost $cost^i(n^*, \bar{n})$ according to equation (19). If not feasible, set $cost^i(n^*, \bar{n}) = \infty$.
14. end for
15. if $cost^i(n^*, \bar{n}) = \infty \forall \bar{n} \in \bar{N}$ then
16. return VDC $i$ is not embeddable
17. else
18. Embed $n^*$ on the node $\bar{n} \in \bar{N}$ with the low $cost^i(n, \bar{n})$.
19. $S \leftarrow S \setminus n^*$
20. end if
21. until $S == \{\emptyset\}$

$$size^i_n = \sum_{r \in R} w^r c^i_{n^r}, \quad (18)$$

$$cost^i(n, \bar{n}) = \gamma_n(mig(n, \bar{m}, \bar{n}) + MigOther(n) + \sum_{n' \in N^i: (n', n) \in L^i} d(n', \bar{n}) \cdot b_{n', n}, \quad (19)$$
Dynamic VDC Consolidation Algorithm

Algorithm 2 Dynamic VDC Consolidation Algorithm

1. Let $\tilde{S}$ represent the set of active machines
2. repeat
3. Sort $\tilde{S}$ in increasing order of $U_{\tilde{n}}$ according to equation (21).
4. $\tilde{n} \leftarrow$ next node in $\tilde{S}$
5. $S \leftarrow loc(\tilde{n})$
6. Sort $S$ according to $size^i_n$ defined in equation (18).
7. for $n \in S$ do
8. $n \leftarrow$ next node in $S$. Let $i$ denote the VDC to which $n$ belongs
9. Run Algorithm 1 on VDC $i$ over $S \setminus \{\tilde{n}\}$.
10. end for
11. $cost(\tilde{n}) \leftarrow$ the total cost according to equation (17)
12. if $cost(\tilde{n}) \leq p_{\tilde{n}}$ then
13. Migrate all virtual nodes according to Algorithm 1
14. Set $\tilde{n}$ to inactive
15. end if
16. $\tilde{S} \leftarrow \tilde{S} \setminus \{\tilde{n}\}$
17. until $U_{\tilde{n}} \geq C_{th}$